

# QDT User Manual

## Version History:

12-12-04	Original	
12-30-04		Fixed SMSQ/E version requirement in appendix
01-23-05	Version 1.00	Added user feedback, updates throughout

## Copyright ©:

2004-2005 James Hunkins [JDH Software Technologies]

## Website:

<http://www.jdh-stech.com>

## Contents

<i>QDT User Manual</i>	<b>1</b>
Contents	<b>1</b>
Documentation Overview	<b>3</b>
Introduction	<b>4</b>
<b>QDT Overview</b>	<b>6</b>
Major Features	6
Objects and Misc	7
Desktop Folder	7
Standard Folder	7
Executable Folder	7
Executable Program	7
Text or Graphic File	8
Unknown File	8
Printer	8
File Management	8
Thing	8
Utilities	9
Tabbed Configuration Notebooks	9
Icon Draw Utility	10
Job Manager	10
Default Icon Manager	11
Theme Manager	11
Installer	12
Integrated Cross System Help	12
Tab LaunchPad	12
Viewers	13
Primary Concepts	13
Objects	13
Data structures and Saving Changes	13
Garbage Collection	14
Saving Desktops	14
Job Ownership	14
<b>Basic Controls</b>	<b>15</b>
Mouse and Keyboard Entries	15

Drop Down Menus	15
Scroll Bars	16
Moving Objects	16
Text Editing	16
<b>Components – Detailed Operating</b>	<b>17</b>
Basic Notes	17
Rename an Object	17
Menu Sensitivity	17
Icon Refreshing	17
Locking Windows	17
Auto Boot vs. Manual Start	18
QDT Icon/Hot Key Access	18
Hotkey Access	18
Desktop Objects	19
Folders	19
Objects within Folders	21
Other Windows	21
Warning/Error	21
User Response	21
About	21
<b>Appendices</b>	<b>22</b>
A: If You Have Problems	22
Contacting JDH Software Technologies	22
B: Software License and Warranty	24
C: Files and Variables	26
Global Variables	26
Primary Files	26
Special Files	26
None	26
D: Limitations	27
E: System Requirements	27
Hardware:	27
Software:	27
F: Additional Manuals and Technical Documents	28
User Manuals	28
Technical and Other Documents	28
G: Other Included Software	28
Licensed	28
Share/Freeware	28
H: Design Philosophy	29
I: Release Plans and Updates	29

## Documentation Overview

The QDT User Manual is designed to supply as much information about how to use QDT and the way it works as possible. A lot of design detail is included to help the user understand some of the design decisions and how particular behaviors actually work.

For a quick start, it is recommended that the user read the QDT Quick Start Guide first which gives an overview of everything from the install procedure through the basics of QDT. After that, this manual will help fill in many of the details, allowing the user to take advantage of many of the additional items not covered in detail in the Quick Start Guide.

In addition to this manual there are others that describe in detail different important components. One that is a must read is the QDT Notebook User Manual. This document gives full details of every notebook information and function available. Notebook details are purposely kept at the minimum in this manual.

This particular manual is laid out in sections which can be jumped to directly. For details, please see the contents at the start of the document. A short highlight of sections and who may want to read them follows here:

Introduction (page 4) – this is a friendly introduction to the whys and concepts behind QDT  
-> not required but should be useful to everyone

QDT Overview (page 6) – this gives details on the structure and architecture of QDT  
-> mostly for the technically inclined. Useful if the user runs into some confusion about why something is done or how something is supposed to be working.

Basic Controls (page 15) – covers how to run controls such as mouse, keys, scrolling, etc.  
-> **everyone** should read this

Components (page 17) – covers all the components and how to use them  
-> **everyone** should read this

Appendix : If You Have Problems (page 22)  
-> if the user has a problem that can't resolve, this is required reading **BEFORE** contacting JDH Software Technologies

Appendix : Software License and Warranty (page 24) – the same as in the Install process  
-> this is what the user is agreeing to by using QDT, **please** read it

Appendix : Files and Variables (page 26) – covers naming and required file suffixes/locations  
-> if the user runs into file conflicts or can't find files, this is useful.

Appendix : Limitations (page 27) – covers demo vs full release limitations  
-> if the user gets 'exceeded' type message, check here for size/count limits

Appendix : System Requirements (page 27)  
-> use to verify that the user's system can run QDT properly

### Other Appendices

-> only for those who haven't had enough yet or might want to thank some of those who have contributed to this effort ☺



Figure 1 : QDT - the modern QL Desktop

## Introduction

The Sinclair QL systems have over many years grown and developed. Today, based on the original powerful but basic multitasking operating system, the QL user community has nearly everything that they could want. With the introduction over the last several years of true vector drawing capabilities, hi-color drivers and the soon to be introduced network access capabilities (my personal conjecture and hopes), there is only one other major piece missing - a true modern desktop environment. This is where QDT - the QL Desktop comes into the picture.

First, it may be appropriate to explain what a desktop is. Think of a physical desktop at home where someone keeps items like:

- papers
- folders of like items such as bills
- a favorite picture or two
- tools such as a calculator, pens and pencils, etc.

Now move this idea to a computer. On the screen one might have:

- a note or two always displayed
- some folders which, when opened, contain documents of some type
- another folder which contains programs

- a folder with programs of a certain type inside it such as a calculator or notepad
- a picture of the user's significant other (or cats as in my case)

Looking at a few of today's systems, the current QL typical desktop (without QDT) can be seen on the left of Figure 2. Directly to the right of that is an image of a typical windows desktop.

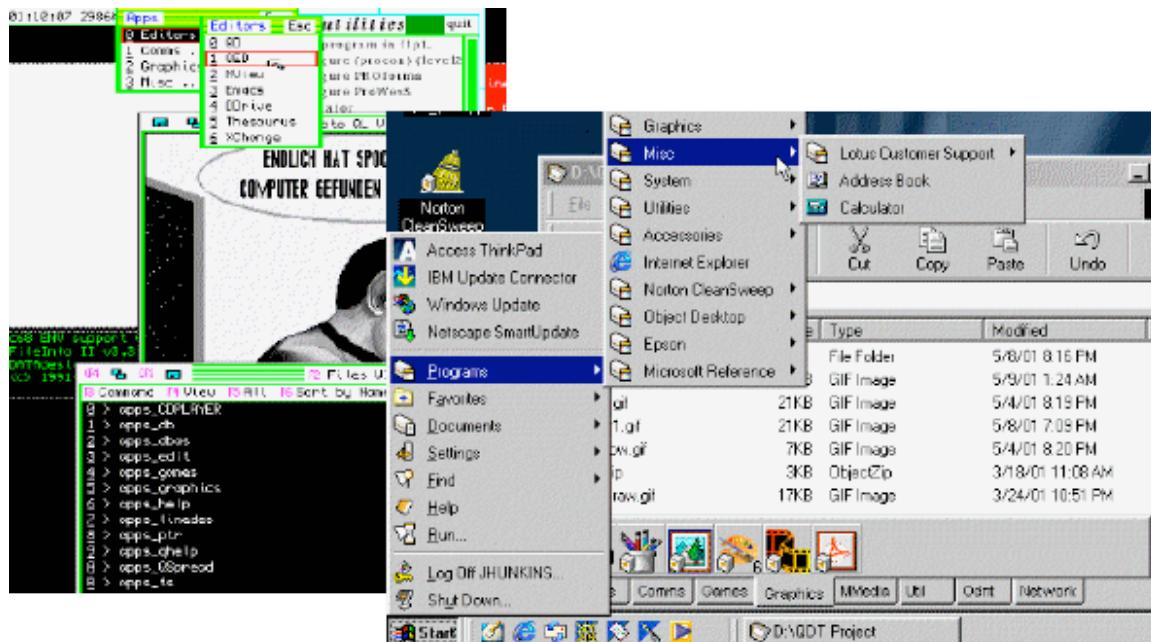


Figure 2 : Desktops – QL Before QDT and Windows 98

Many of the components needed for a desktop do exist in the current QL systems.

#### Windowing Environments

- Pointer Environment
  - tools, pieces, and framework
- ProWesS
  - vectored fonts, message passing

#### Utilities

- File Managers
- File Viewers
  - text and graphics

#### Components

- Disk Directories
- Hot Keys
- Buttons
  - normal and menu
- Q-Menu Pointer Driven Menu/File/Directory Selection
- FileInfo2 File Associations

In the world of PCs and Macs, the computer user finds operating systems with built-in desktops such as Windows 3.1, 98, or XP, Macintosh 9 or X, OS/2, Solaris/X Windows, etc. These all include very comprehensive desktop features and capabilities. But of course they come with a price. I personally find the three worst costs associated with all of them being:

1. the sheer size and complexity of the code
2. the overhead to run the large code base
3. the difficulty to program for them

The concept behind QDT is to use as much of the existing pieces of code on the QL system as is possible and fit a wrapper around everything, adding functionality where needed. QDT takes advantage of several shareware and commercial programs already available to enhance and expand its capabilities and usefulness.

The following sections will outline the major features and components of QDT.

## QDT Overview

### ***Major Features***

The following is a list of major features and concepts included in the QDT architecture and design.

- Folder/Icon Object Based Desktop Environment
  - Desktop - Icons and Folders : both are called "objects"
  - Objects : each object has a prescribed behavior related to different mouse and key strokes
    - Folders - Normal : store other objects including other folders
    - Folders - Executable : same as Normal but can execute a program contained within or elsewhere
    - Executable Files : these can be directly executed
    - Text Files : these can be loaded into a QDT default or user defined program
    - Graphics Files : these can be loaded into a QDT default or user defined program
    - Unknown Files : these can be loaded into a user defined program
- Common Look and Feel
  - Independent override per object/folder
  - Configuration notebook setup for desktop and objects/folders
- PrinterDrag and Drop of Objects : includes file objects onto executable objects (program)
- Independent Windows/Jobs : takes full advantage of SMSQ/E multitasking
  - Each desktop object is its own job
  - Uses SMSQ/E message passing
  - Common items (windows/menus) are independent : can be moved anywhere on screen
- Integration of Existing Software
  - file association : automatic launching of programs by file type
  - drag and drop
  - scratchpad : copy/paste text and images
  - QMenu
- Updated look and feel of QLs to modern standards
- Makes use of hi color and higher resolutions
- Full set of support tools

- Automatic and easy to install and maintain
- Open architecture
  - fully documented
  - make usable for other programs/users

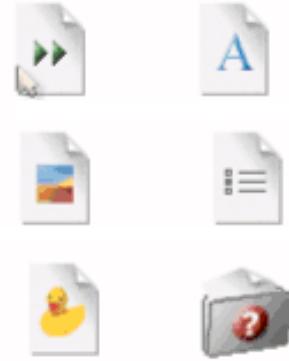
## ***Objects and Misc***

QDT works with what are called objects. An object is simply an item that represents something such as a file, an executable program, a printer, etc. Different types of objects can have different activities and properties assigned to them.

The following is a list of all objects currently being planned.

### **Desktop Folder**

This is the main desktop. It is the same as a standard folder but can't be deleted and always occupies the entire screen. Its background is actually the SMSQ/E background. The desktop has no loose items found in a normal window so all control instead is done through the QDT object's dropdown menu or through the QDT hotkey dropdown menu. All objects sit directly on the desktop as separate programs/windows. Other windows can sit on top of or behind any of its objects.



**Figure 3 : Objects represented as Icons**

### **Standard Folder**

A folder is an object that can contain other objects. It can exist on the desktop or within any folder. It can be opened into a visible folder on the desktop.



It is represented by an icon, which can display whether the folder is open or not, if the user chooses it to do so. This is done by simply defining a different icon to use for when the folder is open versus when it is closed.

It can contain any type of object other than the desktop object, including other folders.

### **Executable Folder**

This is the same as a standard folder but set up to execute an object within it without being opened. When picked, the default mode is that it runs the executable file within it. An optional menu choice is to open it like a normal folder.

These are useful to group things that are used or required by an executable but to also hide them from normal everyday view.

### **Executable Program**

This type of object can be executed. It is basically just an executable file./program.

Command lines can be added to load files when it is launched or access any other functionality that the executable allows with command line parameters.



### Text or Graphic File

These objects represent either text or graphics files. If the object is a text file, then clicking on the object will launch the viewer with the file displayed. For a graphics file, if the viewer understands the format, then the viewer will be launched and the file displayed.



On a object by object bases, an override can be made and a specific program can be set to launch when clicking on the object. Eventually, this capability will be linked into the FileInfo2 capabilities and the user can choose for QDT default viewers, FileInfo2 defaults, or the before mentioned object specific choice.

### Unknown File

These files don't fit neatly into any of the above categories. Clicking on them will not do anything unless a specific program is set to launch for the individual object or the object's file suffix associates it with another executable object.



### Printer

The intent for this object (not yet developed) is to allow the user to have one or more printers, each tied to its own object. After being setup, then another object could be dropped onto the printer object and printed. This could be a file in a native print format or, if a viewer or object specifically assigned program is available, that program would launch and print the file.

This is very preliminary and will likely change before it is ultimately implemented.

### File Management

There has been multiple requests for a File Management type folder, similar to the File Thing that comes with QPac II. The idea here is that this can be used instead of the File Thing or similar File Manager and will be tightly integrated into QDT, allowing dragging and dropping between folders and it.

For example, it would be possible to place a File Management object into a folder for a project which would open the directory within it of all the files for the project. Without this object, individual objects have to be added to a folder. The File Management object would also allow for copying, deletion, etc. of files just like any modern day file management program does.

This also is very preliminary.

### Thing

There might be an special type of object(s) to launch and/or control things with command lines, similar to the capability built into Qascade.

This is just in the initial concept stage and may or may not become reality.

## Utilities

There are many additional pieces/utilities which are used with the QDT desktop. Here is a short list and description of each one. For more information and image captures of each component, go to either the Component – Detailed Operating section for it or its separate User Manual as appropriate.



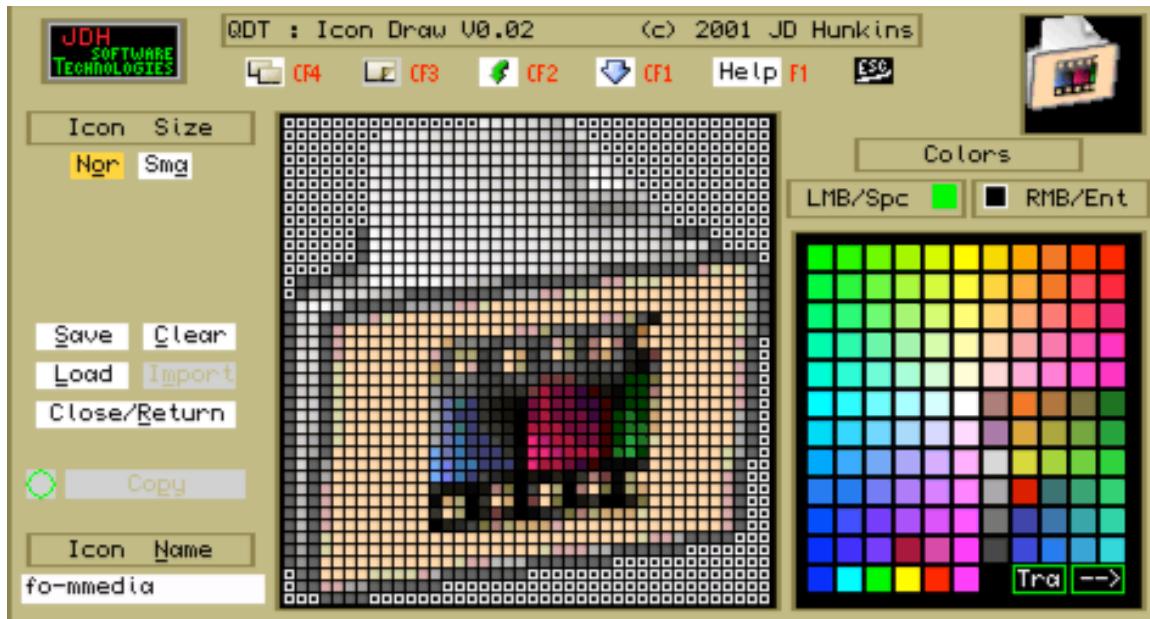
Figure 4 : Tabbed Configuration Notebook

### Tabbed Configuration Notebooks

- Configuration notebooks are used to setup objects and to set/change properties
- They can access
  - QDT system and object properties
  - Config Block Properties
  - Global Properties
  - Program specific properties from a designated file

To handle the large variety of configurable items within QDT and its components, the tabbed notebook approach was taken to manage and present this information. Each type of object has

its own notebook with multiple pages accessed through tabs. Please refer to the separate Users Guide for the Tabbed Configuration Notebooks for details.



**Figure 5 : Icon Draw Utility**

#### **Icon Draw Utility**

- allows a user to draw or modify icons
- allows a user to import icons from different systems/formats

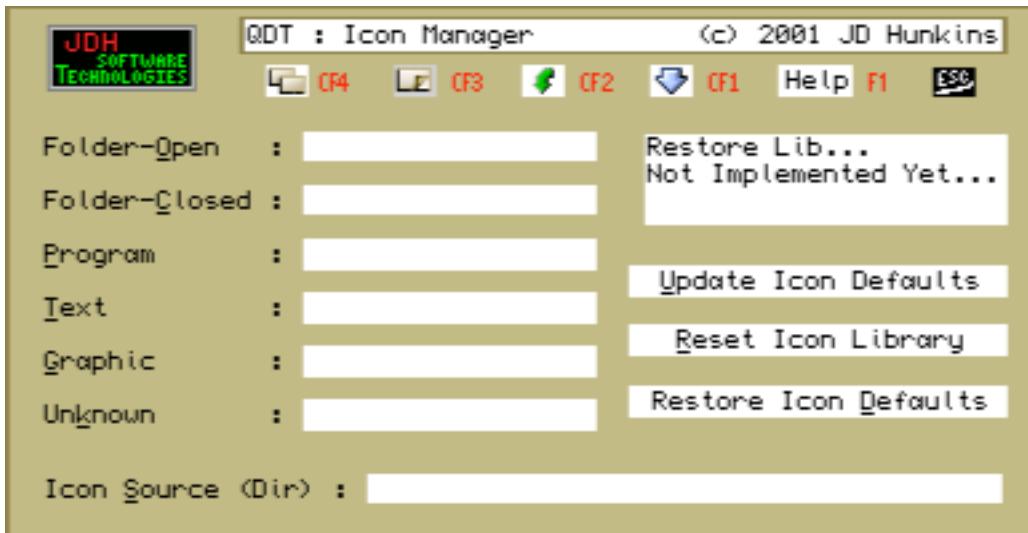
Icon Draw is complex enough and has enough planned enhancements that it gets its own User Manual. Please refer to it separately for details.

There will be a location on the QDT website for sharing of user designed icons.

#### **Job Manager**

- Lists running and sleeping jobs
- Filters can limit visible jobs
- Allows job manipulation
- May have additional functionality such as Hot Key configuration

This utility has not yet been released.



**Figure 6 : Default Icon Manager**

#### **Default Icon Manager**

- restores the default icon set
- allows a user to override the default icon set with their own custom set

QDT comes with its own built-in default icon set, which are used for initial new objects, if an icon is not specified, or if an icon is not found. These are hard coded within QDT as they are critical for the basic operation.

The Default Icon Manager has two functions. Currently it is used by QDT during startup to initialize and include the default icons in the ICONLIB\_QDT binary file which holds the active desktop icons.

It will be possible in the future to override the default icons to some other user preferred icon. The Default Icon Manager will be updated with this functionality included at a later date.

#### **Theme Manager**

- allows designing of a desktop and folder color and background theme
- allows users to import and share their theme designs

There will be a location on the QDT website to allow for sharing of user defined themes.

This utility has not yet been released.

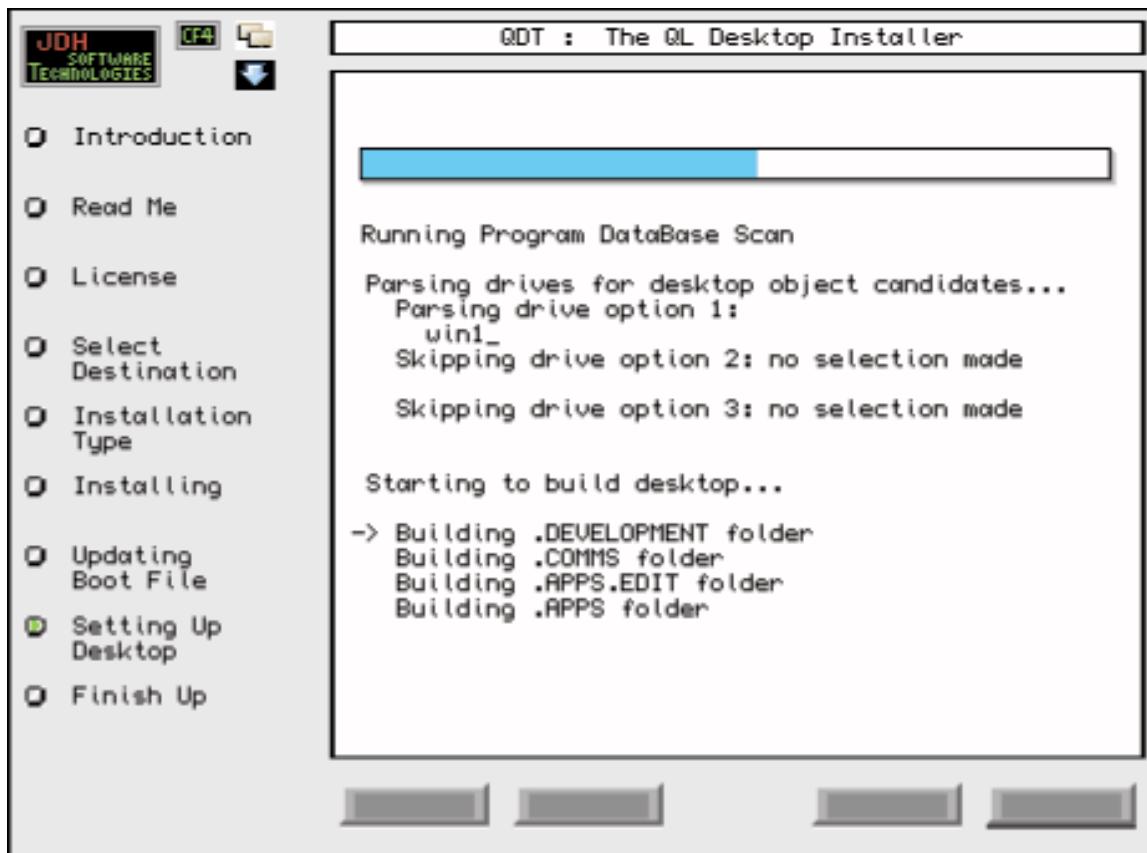


Figure 7 : Sample screen from QDT Installer

#### **Installer**

- Automatically installs or updates the QDT suite of programs
- Can setup a minimal or expanded desktop automatically
- Includes object (program) search and setup

The installer also has its own User Manual.

#### **Integrated Cross System Help**

- Hot key activated help for QDT and individual programs
- Hot link and illustration capabilities built in

This utility has not yet been released.

#### **Tab LaunchPad**

- An alternate way to launch programs
- Organized in tabbed groupings in an object/icon strip

This component should be considered a proposed add-on. Final commitment to produce it has not yet been made.

## **Viewers**

QDT has the capability of calling separate programs to view both text and graphics files. A default viewer is available for both types, along with the overrides per object that can be set in individual object notebooks.

The viewer is simply a user designated program that is launched when a text or graphics object is clicked on, with that related file loaded into the viewer.

## **Primary Concepts**

It is important that the user understands a few basic concepts behind how QDT operates.

### **Objects**

An object is a non-physical thing. It is simply an icon on the desktop or in a folder that refers to some physical action and has related properties. It is even possible for an object to not point to any real file or data, and just contain the default values when it got set up. Of course, that wouldn't be too useful.

Example 1: a text object – this would normally point to a text file somewhere on the disk.

When viewing it or clicking on it, a separate program (either the default viewer or a user assigned one) will open the file for viewing and/or editing or some other purpose.

When this object is deleted or moved, the original file is not affected in any way. It is kind of like a post-it note with the name of the original file and a note as to where the file is. The post-it helps the user find the file quickly. But when the user throws away the post it, the original file still exists (even though they might never find it again).

Example 2: QDT Desktop object – this actually points to not just a file but to a couple of databases. The first is the actual desktop structure (DESKTOP\_QDT which is transferred and accessed from local memory during operation) and the second is the ICON library (ICONLIB\_QDT, also kept in local memory during operation).

Example 3: Normal Folder – this just points to a data structure. It is a piece of the QDT Desktop data structure and can optionally contain objects within it.

If the user tries to delete a folder when there is still something contained within it, QDT will warn them and prevent the folder from being deleted until it is emptied. This avoids the loosing of objects within the folder by mistake.

### **Data structures and Saving Changes**

QDT keeps two primary sets of data plus several arrays of text and pointers. When it is not running, data structures are kept on the disk only. When QDT starts up, it loads the physical structure database kept in DESKTOP\_QDT into memory in a binary form for quick access and manipulation.

The second structure is the ICONLIB\_BIN file. Upon startup, QDT rebuilds this database from the DESKTOP\_QDT file and the individual \_ICON files referred to in the desktop file. This structure stays in the disk file during operation. When building this structure, QDT sets up a list that points into the file for each icon that it may need and only loads the icon into memory from

the structure as needed. This is very fast due to the small icon size plus the fact that QL caches files in memory that it has accessed as long as there is enough memory to do so.

### ***Garbage Collection***

At the point, QDT only partially reuses space when something gets deleted. However, this type of activity isn't that common per a single session and limits are high enough that it was decided to not perform live garbage collection during a session. Garbage collection which would tend to slow down the system and potentially add a lot of extra complexity.

The current design does begin with all structures clean every time QDT is restarted. If, however, QDT was to be left running for very long extended periods of time and a lot of text changes, object deletions, etc. occurred, it might eventually run into some internal limits, of which the user would be warned. If this ever does happen, simply restart QDT.

### ***Saving Desktops***

The internal binary structure of the DESKTOP\_QDT file introduces the concept of a 'local' desktop. Basically, any change done on the desktop is not permanent, unless the user physically tells QDT to save the desktop to disk. This allows the user to change their desktop for a specific session but keep the original desktop image intact for next time.

Upon quitting, if QDT sees that changes have been made locally but not saved to the disk, it will ask if it should save the current desktop.

QDT does offer the options to archive desktops and recover them, up to 4 at a time. Please refer to the Desktop Notebook section of the Notebook User Guide for information on how to use this.

Eventually, an additional ability will be available to allow for the maintaining of multiple desktops and switch between them between sessions. and even during a session (it will require a restart). This will be handy if different users use the same system (each can have their own desktop) or if the user has different types of activities/projects that the system is used for. Individual desktops can be customized for the specific uses, helping to improve workflow efficiency on the computer.

A limited version of this capability is currently available from the Desktop notebook.

### **Job Ownership**

When a program is launched from within QDT, it is owned by QDT and not by job 0, as would be the case when launching a program from SuperBasic. This means that if a program is running and QDT is exited (or heaven forbid, crashes), the program will also close and potentially lose any unsaved data.

To prevent accidental data loss, when exiting QDT, the desktop will check to see if there are any open programs owned by QDT. If it finds any, it will let the user know and give an option to abort the quit until the programs have been gracefully exited.

On the other hand, if everything has been saved from the programs and the user doesn't have any concerns about file corruption or other potential issues, then they could always choose to just quit and let all the programs conveniently get closed for them.

CAUTION: some programs keep files open while they are running and potentially could corrupt those files if closed forcefully. In other words, the user should allow the forced closing at their own discretion/own risk.

## Basic Controls

There are multiple ways to trigger QDT to do things. They all include the mouse and/or keyboards but have visibly different mechanisms, as described in this section.

### Mouse and Keyboard Entries

The following table shows the response to different mouse and keyboard entries, depending on where they occur. RMB stands for Right Mouse Button (alternate is 'RETURN' key), LMB stands for Left Mouse Button (alternate is 'SPACE' key), and TEXT is direct text entry. If a shortcut key is available, it has the same response as RMB does.

Note: most shortcut keys are only active if the cursor is in the window where the shortcut key is displayed.

**Table 1 : Key and Mouse Entry Actions**

Location	Entry	Response
Icon	QDT	LMB
Icon	Desktop or Folder	LMB
Icon	All	RMB
Icon Text	QDT	LMB
Icon Text	Desktop Or Folder	LMB
Icon Text	All	RMB
Folder	Empty area	LMB
Folder	Empty area	RMB
Desktop	Empty area	LMB
Desktop	Empty area	RMB
Desktop	Any	HotKey

### Drop Down Menus

Drop Down menus are used extensively throughout QDT to give multiple options. Depending on the location that the menu is called from, the selections vary depending on what actions are available in that particular situation. Figure 8 shows two different menus, illustrating the context sensitivity of each menu usage.

All menu selections can be picked by either a LMB or RMB selection or by pressing the key that is underlined within a menu entry.

To quit a menu without selection anything, simply press the ESC key or click on the ESC icon.



**Figure 8 : Context Sensitive Drop Down Menus**

## **Scroll Bars**

If an open folder has more objects than can be displayed, then either vertical and/or horizontal scroll bars are displayed within the folder window. The width of the darker section is directly related to the visible space versus the space left that would be required to display the other objects which are outside of the folder window.

Clicking to either side will redraw the window one standard row or column offset from its original position. If clicked on the solid part in the middle, no response will happen. A standard offset is the spacing used by QDT when adding new objects to a folder or the desktop.

Resizing the folder so that all objects are visible will automatically remove the scroll bars.

## **Moving Objects**

While moving objects on the desktop, the is in some cases controlled by the MOVEMODE which is set with the SuperBasic command line:

WM\_MOVEMODE n

Value = 0 : the standard QL move sprite is used

Value = 1 or 2 :

if CTRL F4 or picking 'M' from the menu, standard QL move sprite is used if clicking on Move from the drop down menu and holding the mouse button down, then the outline (n=1) or the full icon/text (n=2) is displayed during the move, allowing for more accurate placement

Within a folder, the full icon image less the text is always visible during an object move, regardless of how the move is initiated.

The first release of QDT does not support object moving between folders. This will be enabled in a future release.

## **Text Editing**

When changing labels or filling in text boxes within notebook, direct text editing is done with most of the standard QL editing keys, along with the standard characters. The following table outlines the special function keys used for text editing.

**Table 2 : Text Editing Keys**

Left	move cursor left one character
Right	move cursor right one character
Alt Left	move cursor to the first character on the left
Alt Right	move cursor to the last character on the right
Delete	deletes character under the cursor
Back Space	deletes character to the left of the cursor
Shift Left	moves cursor to first non ASCII character to the left such as an '‘, or '‘
Shift Right	moves cursor to first non ASCII character to the right such as an '‘ or '‘
Enter	exits edit mode and saves the edited results
ESC	exits edit mode and restores the original entry, discarding any edits done

Text label editing starts in insertion mode. Notebook text editing is always in overwrite mode.

Object labels can be up to three lines in height at up to 16 characters per line. To force a line return use the characters '‘n'. These two characters will not be displayed and do not count as

part of the 16 characters per line limit. They instead will force a line return when the text is displayed on a notebook or on the desktop. The two characters will be displayed in the notebook for reference and editing.

## Components – Detailed Operating

### **Basic Notes**

#### **Rename an Object**

Renaming can be achieved by either choosing ‘Rename’ in the drop down menu or by right mouse clicking directly onto the object’s text field.

A separate window is opened which displays the current text. Most standard QL editing controls are supported.

To enter a line return, use the two characters ‘\n’ . Up to three lines of 16 characters of text per object name is allowed. If a line exceeds 16 characters, it will get wrapped to the next line on the screen. If an ‘accidental’ wrap around occurs and extra line returns are used, forcing 4 or more lines, or if there are 3 or more line returns, the top lines will scroll out of the text region and not be visible.

#### **Menu Sensitivity**

Commands from a drop down menu affect different things, depending on from where the menu is dropped.

- commands from a folder menu (within the folder or from the folder’s icon) : affects the folder directly
  - EX: Adding an object adds it to the folder itself, not to the desktop or anywhere else
  - EX: Deleting will delete the folder itself
- commands from the QDT icon or the hotkey menu affects the desktop in general
  - EX: adding an object here places it directly on the desktop
- commands from a non-folder object just affect that object

#### **Icon Refreshing**

Unfortunately, with the current PE capabilities, if a window is moved out from under an object’s desktop icon that has transparent pixels, it still isn’t possible to avoid garbage left behind in the icon’s transparent sections. To clean this up, choose refresh from any Desktop object which will cause the desktop to redraw all its icons, cleaning up the transparent sections of each.

This is not an issue within a folder as the icons are drawn directly within the folder window. No other windows can get between them. Refresh from a folder menu does refresh the icons within that folder, just in case it does somehow get corrupted.

Methods to monitor window stacks are being thought about but this will be fairly complex and is considered a low priority enhancement for a potential future release.

#### **Locking Windows**

QDT in several cases will lock a window down so that it can not be picked. This occurs most commonly when a window calls a separate window, such as a drop down menu or a notebook. The original window is unlocked when the called window is closed.

If the user changes their mind and doesn't want to do an entry in one of these called windows, in most cases they can do an ESC to get back to their original calling window.

### **Auto Boot vs. Manual Start**

The installer, when the Easy install or Advanced install with building a boot file are chosen, assumes that QDT should autoboot every time the QL is started. It therefore places the following line into the boot file (assuming the QDT location is win1\_QDT\_):

EX win1\_QDT\_BIN\_QDT\_EXE

If autoboot to QDT is not desired, simply remark this line out. QDT can then be manually started after the first reboot by going to the QDT binary directory and typing:

EX QDT\_EXE

The QDT binary directory is the QDT main directory that was designated plus 'BIN\_ '.

The user must, in either case, make sure that all the QDT specific global variables plus any other recommended boot file changes have been included before starting QDT, which is why a reboot is normally required.

### ***QDT Icon/Hot Key Access***

The primary control for QDT is either through its Desktop Icon or through its HotKey menu access. A drop down menu gives the options shown in Table 3.

All these functions work directly on the desktop. The properties will bring up the desktop notebook which controls the overall desktop functionality plus many of the standard QL system activities such as screen resolution, wall paper, and screen savers.

One of the QDT configuration options that is available is to not display the QDT object on the desktop. If this is done, then the only way to access this menu will be through the HotKey method.

**Table 3 : QDT Menu**

Properties	P
Move	M
Refresh	S
Add new Object	A
Add new Folder	F
Save Desktop	V
Quit	Q
About	A
Help	H

**CAUTION:** do not disable the QDT icon display if the hotkey option is not activated. The only way to quit QDT in that case will be to force quit it. Adding objects to the desktop will also not be possible.

### **Hotkey Access**

For the HotKey access, all the same functions are available except for the Move selection.

During the install process, if every HotKey was not already used (very unlikely), QDT will have assigned a hot key . This hot key will call a program that generates a QDT drop down menu from anywhere on the desktop. The user does not have to find the QDT icon to do this. Primary commands such as Quit, Restore, Save, and Add are included for easy and quick access (see Table 3 for the full list of commands).

If the Advance install option is used, the user picked their own key from the list of available ones. If Easy install was used, the installer automatically picked the first available key in its priority list that it could find (see the Installer User Manual for more information).

If QDT is not running but the user tries the HotKey, the menu program will run. But the program will see that QDT is not present and simply quit itself without doing anything.

### **Desktop Objects**

Desktop objects also have drop down menus that vary their content depending on the type of object and its state.

Table 4 shows the normal desktop file object menu for both text, graphic and unknown files. Note that if an associated program is not assigned to an unknown type object then the View option will not actually do anything.

Refresh in all desktop object menus will affect the desktop icons.

### **Table 4 : Desktop File Icon Menu**

View	V
Properties	P
Rename	R
Move	M
Refresh	S
Delete	D
About	B
Help	H

### **Table 5 : Desktop Folder Icon Menu**

Open	O	If closed
Close	C	If open
Execute	E	Only for exec type
Properties	P	
Rename	R	
Move	M	
Resize	Z	Only if open
Refresh	S	
Delete	D	
Add New Object	A	
Add New Folder	F	
About	B	
Help	H	

Table 5 shows the desktop folder menu. The entry is dependent on whether the folder is open or not. The Execute option only shows up for an Executable folder type. The default click option for an Executable folder is to Execute the designated program and not to open or close the folder itself. The only way to open an executable folder is through its drop down.

The resize option is only available if the folder is open.

Most of the entries in this folder menu will directly affect the folder. For example, if Add New Object is chosen, then it is placed within the folder, even if the folder is not open at the time.

The one exception to this is the Refresh command which refreshes the desktop objects. To refresh the folder's objects, the folder's menu from within the open folder or a loose item's (within the opened folder) menu must be accessed.

As was mentioned previously, clicking on a normal folder object which is already open will pick that folder to the top. This is a convenient way to dig out a buried folder if there is too much clutter on the desktop.

### **Folders**

Folders can hold any type of object. A typical folder with its drop down menu activated is shown in Figure 9.

An object icon can be added and placed anywhere within a folder. They can even overlap. However, which one is selectable is not well defined so this is not recommended.

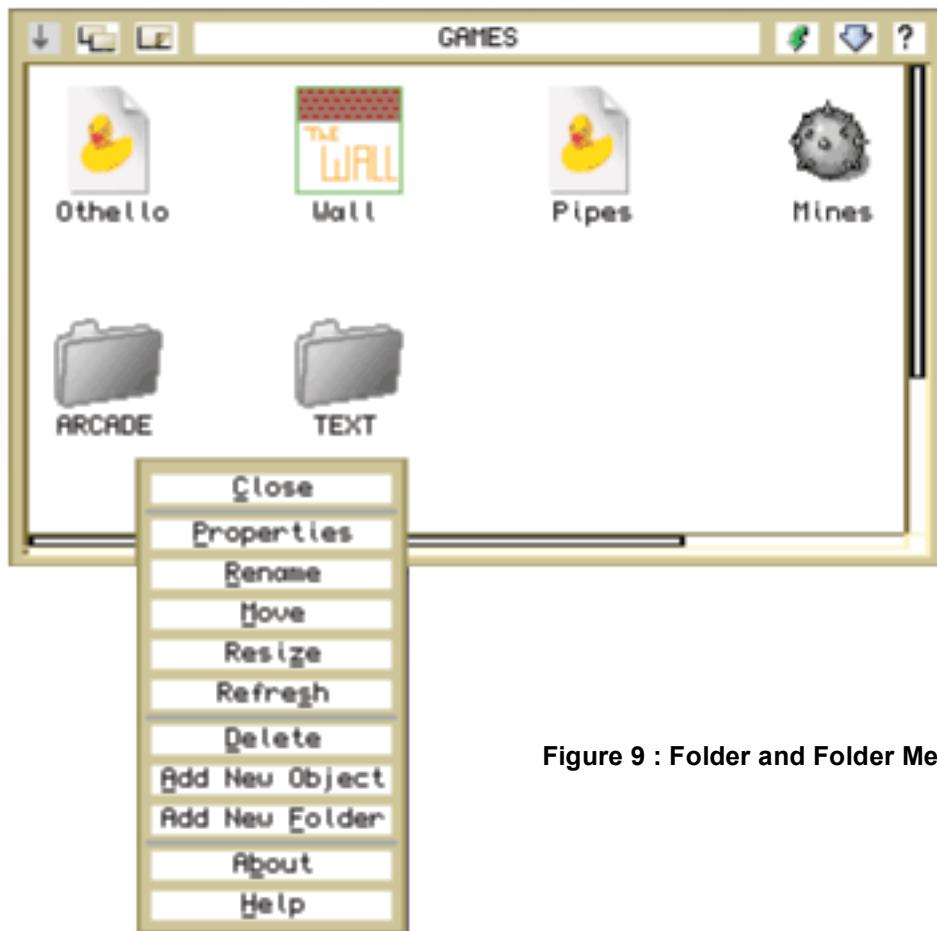


Figure 9 : Folder and Folder Menu

For control, open folders are made up of different control areas:

- Top Loose Items
- Optional Scroll Bars
- Open Folder Area
- Object Icons

Table 6 shows the loose items and their function, going from left to right. For the most part they are standard pointer environment controls. There are a few variations to the standard behavior as follows.

- **Resize** : right or down shrinks the window. Up or left enlarges it.
- **Title** : picking this allows the title to be edited without opening the folder's notebook.
- **Refresh** : on a sufficiently fast system, unless there has been folder display corruption, the results of this selection may happen too fast to be visible
- **Close** : many users may think of this as a sleep icon and would basically be correct. However, instead of drawing a separate button, the folder just lets its parent object folder icon know that it is closing so that the object can redraw its icon to the defined closed icon image. After that the folder physically closes itself and quits its local executable (an open folder is actually a separate program that is owned by QDT).

Table 6 : Folder Loose Items

Item	Short Cut
Menu	F3
Move	CNT F4
Resize	CNT F3
Title	F2
Refresh	CNT F2
Close	none
Help	F1

The scroll bars are only displayed if there are objects that aren't visible because the window is too small. If the folder window is enlarged enough the scroll bars will be removed.

The open area between objects is considered the RMB menu selection area. Doing a RMB in this area will pull up the same drop down menu that the menu loose item or the parent folder icon generates. See Table 5 for a list of menu contents (same as a folder on the desktop).

### **Objects within Folders**

As on the desktop, objects are represented by icons within folders. By clicking on the icons or the text underneath them, different functionality occurs as described in the Table 1 : Key and Mouse Entry Actions, in the Basic Controls section.

The drop down menus in this case are the same as those gotten with the desktop objects, as shown in Table 4 and Table 5.

### **Other Windows**

There are several other small windows used by QDT for communications with the user. The three most common are:

#### **Warning/Error**

The warning/error reporting window will display a W for a warning or an E for an error. A warning is something that is non-critical. An error is something that has caused an important operation to abort. The top message is a category level quick response. The larger window gives additional details as appropriate.

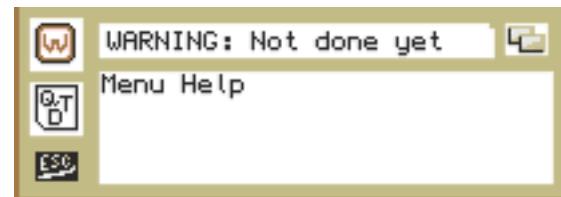


Figure 10 : Warning/Error Message Window

#### **User Response**

This kind of window will either display an informational message with a single OK button or ask a particular type of question, with 2 or more buttons to choose from. Short cut keys are available for any button with an underlined letter which represents the actual key to be used. The current operation will not continue until a button is pressed or ESC is hit.

If an escape is entered, this will be treated as a No or Abort response.

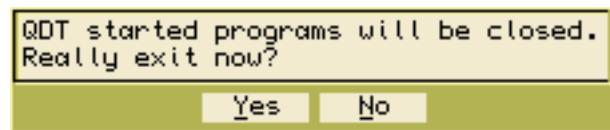


Figure 11 : User Reponse Window

#### **About**

The about window gives fundamental information about QDT, including versions numbers, release notes, etc.



Figure 12 : About Window

# Appendices

## **A: If You Have Problems**

If you run into any issues and/or problems, the following are suggested methods to approaching the understanding and resolving them.

- double check the user manual to make sure that you understand how the feature that you are trying to use works.
- check the included readme files for known issues, work arounds, and unimplemented features
- check on the QDT website [ <http://www.jdh-stech.com> ] for:
  - updated lists of issues, etc.
  - trouble shooting assistance
  - software updates
- check the QL email users lists or your local user group for assistance
- check with your QDT distributor for software updates
- contact JDH Software Technologies by email for assistance at:

[contact@jdh-stech.com](mailto:contact@jdh-stech.com)

NOTE: Please do not include this email address in any forum email or websites (an attempt to avoid turning it into a spam magnet). Always check the website for updated email addresses, etc.

### **Contacting JDH Software Technologies**

When contacting JDH Software Technologies for assistance, it is very important that you **ALWAYS** follow the following guidelines to avoid delays and to help us support you better

- 1) Include in the subject line 'QDT support'
- 2) Cut and paste the following form (or directly enter it) into the email body:

#### QDT Info

QDT Purchaser (your name) :

QDT Serial Number :

QDT Version Number :

{ get this from the About selection in the Desktop drop down menu }

Component Involved if known :

{ open folder, icon on desktop, icon in folder, drop down menu, notebook, etc }

Detailed description of problem :

What steps did you go through to make it happen?

Is it reproducible?

What steps have you tried to work around it?

System info

System Type :  
(QPC, Aurora, Q40, Q60, etc)

QPC Version (if applicable) :

SMSQ/E Version :

Hard drive size :

Memory size :

3) Include the following files:

if an install or update problem:  
QDTinstall\_log  
DESKTOP\_QDT (if it generated it)

if not an install problem  
DESKTOP\_QDT  
qdt.log

## **B: Software License and Warranty**

The following is a direct copy of the license that is included with the QDT package. It can be read from within the QDT installer where the user agreement is required before the installation can be continued. The text can also be found in the QDT DOCS\_ directory as "license\_.txt".

=====

QDT (QL Desktop) License  
All Official Full Release Versions

Copyright © 2004-2005 James D Hunkins  
JDH Software Technologies  
-> All rights reserved.

This is a legal agreement between you and JDH Software Technologies (James Hunkins) covering your use of QDT and any programs included with the original package (the "Software"). Be sure to read the following agreement before using the Software.

**BY USING THE SOFTWARE (REGARDLESS IF YOU HAVE REGISTERED THE SOFTWARE OR NOT), YOU ARE AGREEING TO BE BOUND BY THE TERMS OF THIS AGREEMENT. IF YOU DO NOT AGREE TO THE TERMS OF THIS AGREEMENT, DO NOT USE THE SOFTWARE AND DESTROY ALL COPIES IN YOUR POSSESSION.**

The Software is owned by JDH Software Technologies and is protected by United States copyright laws and international treaty provisions. Therefore, you must treat the Software like any other copyrighted material (e.g., a book or musical recording).

Redistribution of this release package in any form is not permitted. Distribution is only allowed through authorized distributors as per agreements made exclusively with JDH Software Technologies.

Use of this package is permitted by the original purchaser only for their own personal use. Separate copies can be put onto different computers as long as only one copy is in use at one time (the book usage concept) and all copies are only used by the purchaser of this license and/or his immediate household or organization.

Distribution of any additional included software not developed by JDH Software Technologies is controlled as per documentation included with all such software.

Any usage outside of this described model must receive specific written authorization from JDH Software Technologies.

The Software and its related documentation are provided "AS IS" and without warranty of any kind and JDH Software Technologies expressly disclaims all other warranties, expressed or implied, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. Under no circumstances shall JDH Software Technologies be liable for any incidental, special, or consequential damages that result from the use or inability to use the Software or related documentation, even if JDH Software Technologies has been advised of the possibility of such damages. In no event shall JDH Software Technologies' liability exceed the license fee paid, if any.

This Agreement shall be governed by the laws of the State of California. If for any reason a court of competent jurisdiction finds any provision of the Agreement, or portion thereof, to be unenforceable, that provision of the Agreement shall be enforced to the maximum extent

permissible so as to effect the intent of the parties, and the remainder of this Agreement shall continue in full force and effect.

This software package may include licensed code from other developers as noted below along with voluntary contributions made by many individuals. For more information on JDH Software Technologies, refer to the website "<http://www.jdh-stech.com>".

Graphics/Color Development Support:  
Thorsten Herbert, Italy

Additional Icons By:  
Roy Wood, England

=====

Included licensed code in non-demo version : included in the QDT pricing

QMenu menu extensions  
Copyright© 1989-97 Jochen Merz Software

=====

Included Non-licensed code : no charge distribution

FI2config\_obj FileInfo2  
FileInfo2\_bin  
FileInfo2\_txt  
FI2config\_help  
Copyright © 1993-1999 Thierry Godefroy

MenuConfig 02  
Copyright © 1992-2003 Jochen Merz Software

## ***C: Files and Variables***

### **Global Variables**

QDT_LOC	QDT base directory, location of QDT desktop database file
QDTICON_LOC	QDT icon directory
QDTICONLIB_LOC	QDT binary icon library database directory
QDTBIN_LOC	QDT executable file directory
QDTTHEME_LOC	QDT theme file directory
QDTBACKGND_LOC	QDT background file directory
QDTDOC_LOC	QDT documentation directory
QDTHHELP_LOC	QDT help file directory

### **Primary Files**

Desktop_QDT	QDT desktop primary database { do NOT hand edit }
DeskTop_xxxxxxxxx_QDT	Archive of QDT database xxxxxxxx is an 8 digit hex number giving a time stamp in seconds (standard QL time)
ICONLIB_QDT	QDT binary copy of Icons in use by current desktop
xxxx_ICON	QDT icon file
xxxx_THM	QDT desktop themes file
xxxx_STUP	QDT startup file

### **Special Files**

None

#### **D: Limitations**

**Table 7 : QDT Demo vs. Full Install Limitations**

	<b>Full Install</b>	<b>Demo</b>	<b>Notes</b>
<b>Time</b>	Unlimited	30 days	From date of install
<b>Total Objects</b>	500	15	All desktop objects, including QDT as one
<b>Total Folders</b>	100	5	Total folders in the desktop
<b>Objs in a Folder</b>	100	5	Total objects within any one folder

#### **E: System Requirements**

For any capabilities this advanced, there will be minimum system requirements. Here are the QDT hardware and software requirements that are anticipated. Please note that the minimum requirements are just that - minimum. Many of them will impact the quality of the user's experience with QDT but at least it will run.

##### **Hardware:**

**Table 8 : QDT Hardware Requirements**

<b>Configuration</b>	<b>Optimal</b>	<b>Minimum</b>
<b>Drive Type</b>	Hard Drive	Flash ROM Drive <sup>*1</sup>
<b>Drive Size</b>	?	?
<b>Memory</b>	4 Mbytes+	?
<b>Resolution</b>	800x600 +	512x256 +
<b>Color Mode</b>	Hi Color	Hi Color

Notes:

1: not yet supported

##### **Software:**

**Table 9 : QDT Software Requirements**

<b>Type</b>	
Operating System	SMSQ/E v3.07 or higher
Windowing System	Pointer Environment
Included Software	QMenu
Freeware/Shareware	FileInfo2 Env_bin

## ***F: Additional Manuals and Technical Documents***

As of the latest update of this manual, the following documents are either available or in preparation.

### **User Manuals**

QDT Quick Start Guide

QDT Installer User Manual

QDT Notebook User Manual

QDT Icon Draw User Manual

### **Technical and Other Documents**

QDT Icons Technical Reference

## ***G: Other Included Software***

### **Licensed**

QMenu            Copyright © 1992-2003            Jochen Merz Software  
- this is only included with the full release, not the demo

### **Share/Freeware**

FileInfo2	Copyright © 1993-1999	Thierry Godefroy - documentation included in package
Env_bin	no copyright found Part of QDOS C68 distribution	Original Version: Dave Nash C Code Modification : Dave Walker SuperBasic Interface : Dave Woodman
unzip	Copyright © 1990-1999 QL Version developed by	Info-Zip Jonathan R Hudson - documentation included in package

## ***H: Design Philosophy***

Optimize for high color, higher resolution

- back design for lower

Make usable to wide range of users

- but don't cripple to support minimum user scenario

Reuse and integrate as much of today's existing software as possible

- take full advantage of built-in and available capabilities

Optimize for performance and code size

- balance

## ***I: Release Plans and Updates***

The code release will be in phases. Here is a list of how it is expected to occur. The initial column will be the first release. Then, in approximate priority, the items and support in the follow-up column will occur. It should be noted that, even though some of the support will not be initially available, the architecture of QDT and code development is being done to ensure a smooth addition of capabilities during the release process.

**Table 10 : QDT Release Plans**

<b>Initial</b>	<b>Follow-up</b>
Hard Drive Required	Flash-ROM support
High resolution and color optimization	Full Color Palette
Base package <ul style="list-style-type: none"><li>• Basic Desktop (Folders and Objects)</li><li>• Installer for QDT</li><li>• basic Tabbed Config Notebook<ul style="list-style-type: none"><li>- not all features will be enabled</li></ul></li><li>• Icon Draw</li></ul>	Add-ons <ul style="list-style-type: none"><li>• Drag and Drop</li><li>• Tabbed Config Notebook<ul style="list-style-type: none"><li>- all features fully enabled</li></ul></li><li>• Job Manager</li><li>• Objects<ul style="list-style-type: none"><li>- Printer</li><li>- FileManager</li><li>- Thing</li></ul></li><li>• Theme Manager</li><li>• Icon Browser</li><li>• Tab LaunchPad (optional)</li><li>• Integrated Cross System Help</li><li>• Enhanced Icon Draw</li></ul>

The major update mechanism being planned is through internet downloads and through the normal distributor floppy drive update methods. The QDT installer is being enhanced to add updating capabilities without requiring a full re-install each time.

Through 2005 all updates will be at no charge as additional functionality is added. The final functionality level at the end of 2005 will depend on the actual interest in QDT. Note that there is a minimum set that is expected to be implemented, regardless.